REMARKS

This application has been carefully considered in connection with the Examiner's Office Action dated March 28, 2007. Reconsideration and allowance are respectfully requested in view of the following.

Summary of Rejections

Claims 1-43 were pending at the time of the Office Action.

Claims 1-43 were rejected under 35 USC § 101 because the claimed invention is directed to non-statutory subject matter.

Claims 24-26 were rejected under 35 USC § 112, second paragraph as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 1-43 were rejected under 35 USC § 102(a) as being anticipated by "BEA Tuxedo®: Using the ATMI /Q Component," January 2003, BEA Systems, Inc. (hereinafter "BEA").

Claim 1 was objected.

Summary of Response

Claims 1, 2, 16, 17, 23-30, and 43 were amended.

Claims 3-15, 18-22, and 31-42 remain as originally submitted.

Remarks and Arguments are provided below.

Summary of Claims Pending

Claims 1-43 are currently pending following this response.

Response to Objection

Claim 1 was objected to because of an informality. Claim 1 has been amended to read "the system comprising" as suggested in the Office Action.

Applicant Initiated Interview

On June 14, 2007, Applicant initiated a telephone call to Examiner Kiss in an effort to request an interview of the pending application. In the telephone conversation Applicant and Examiner Kiss agreed to delay the interview until after the end of the quarter and subsequent to filing this response. Applicant respectfully requests an interview with the Examiner upon receiving and reviewing this response.

Response to Rejections

The disclosure is directed to addressing the limitations of the COBOL programming language. Paragraph 006 of the disclosure states:

> "Unfortunately, COBOL is severely limited in a number of areas compared to the processing techniques available to developers that use other languages such as C or JAVA. POSIX, or Portable Operating System Interface uniX, is a standard UNIX interface for applications to ensure interoperability on equipment from various venders. POSIX includes well know functionality available in programming languages such as C and JAVA for accomplishing distributed and asynchronous processing, such as shared memory, memory and message queues, threads, semaphores and mutexes, events, signal handlers, and sockets."

Further, paragraph 008 of the disclosure states:

> "The processing techniques described above are examples of useful functionality widely available to programmers using distributed and asynchronous processing languages, such as C and JAVA, but unavailable in COBOL. Frequently, it is desirable for business processes employing COBOL applications to accomplish distributed and asynchronous processing. Although the COBOL language has limitations, it is difficult for businesses with a significant investment in COBOL programs to justify abandoning the COBOL applications and redeveloping the applications using a more modern and flexible language, such as C or JAVA. Instead, COBOL systems are typically provided with an interface or "hook" to enable the COBOL program to cooperate with, for example, C or JAVA programs. The C or Java program then performs the distributed and asynchronous processing tasks that the COBOL application is otherwise incapable of handling independently."

The disclosure utilizes a technical layer that defines bit level mappings of calls to functions of an operating system so as to interface with the operating system in order to enable COBOL programs to perform tasks that are not native to COBOL. In an embodiment, the technical layer may be implemented as a library of callable routines that are linked to the COBOL programs. In other embodiments, the technical layer may be implemented as a pre-compiler or a compiler, for example. The technical layer is described in detail in paragraphs 025-038 wherein a discussion of the bit level mapping of calls is discussed in paragraphs 040-045.

With the technical layer defining how to interface with functions of the operating system, COBOL programs may employ the technical layer in the same runtime environment such that additional interfaces with other high level programming languages are not necessary. This enables COBOL programs to be executed in their native COBOL runtime environment so as to take advantage of the inherent efficiencies of COBOL programming while also being able to perform tasks that are not native to COBOL.

The disclosure utilizes a shared memory routine 20c, a memory queue routine 20g, and a message queue routine 20h of the technical layer to provide COBOL programs with functionality to enable memory sharing, memory queues, and message queues (see paragraphs 035 and 036 of the disclosure). For example, as shown in FIG. 6 and described in paragraphs 067-075 of the disclosure, an address 94 of a block of memory 36 may be retrieved by a COBOL program 12 through the memory queue routine 20g.

An operating system 34 may maintain the address 94 and a key 92 related to the address 94. The COBOL program 12 may make a linkage call with an identifier to the memory queue routine 20g. The memory queue routine 20g may maintain an index 90 including the identifier and the key 92 associated with the identifier. The memory queue routine 20g may look up the identifier and obtain the key 92. The memory queue routine 20g may communicate with the operating system 34 to retrieve the address 94 based on the key 92. The memory queue routine 20g may return the address 94 to a linkage section 96 of the COBOL program 12. The linkage section 96 is a standard portion of a COBOL program that is ordinarily used to receive data that is passed in from other programs. By mapping the linkage section of the COBOL program 12 to the address 94, the COBOL program 12 resolves the data at the address 94 of the memory 36 to the linkage section 96 of the COBOL program 12. Thus, the COBOL program 12 considers the information stored in memory at the address 94 as local and accessible to the COBOL program 12.

Similarly, as shown in FIG. 7 and described in paragraphs 076-083 of the disclosure, the shared memory routine 20c may provide COBOL programs with

functionality to enable memory sharing. The functionality to enable memory sharing with COBOL programs may be provided through a COBOL program 50 or 52 obtaining an address 114 using a key 112, an index 110, an identifier, and a linkage section 96a or 96b, similar to the memory queue routine 20g described above.

BEA2003 discloses an application-to-transaction monitor interface (ATMI) that provides functions that allow messages to be added to or read from queues (page 1-1). BEA2003 provides support for ATMI COBOL language functions for enqueuing and dequeuing messages (page 4-1). While BEA2003 generally teaches enabling queues for COBOL programs, the queues are enabled using the ATMI, which is a C programming interface. Providing queue functions in COBOL using the ATMI is similar to how other COBOL systems are provided with an interface or "hook" with which a COBOL program cooperates, as discussed in paragraph 008 of the disclosure. BEA2003 also generally discloses shared memory through the application queue space as illustrated in Figure 1-1 on page 1-2 and through the storage used in peer-to-peer communications as illustrated in Figure 1-2 on page 1-5. Applicant respectfully submits that BEA2003 does not disclose that the ATMI enables the queue functions or shared memory in COBOL using the various structures such as the key, the index, and the linkage section of a COBOL program as recited in the claims.

Response to Rejections under Section 112

In the Office Action dated March 28, 2007, claims 24-26 were rejected under 35 USC § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 24 has been amended as interpreted by the Office Action to depend from claim 23. Applicant respectfully requests that the 35 USC § 112, second paragraph, rejection of claims 24-26 be withdrawn.

Response to Rejections under Section 101

In the Office Action dated March 28, 2007, claims 1-43 were rejected under 35 USC § 101 because the claimed invention is directed to non-statutory subject matter.

Claim 1:

Claims 1-15 were rejected as elements that can be reasonably interpreted as software, wherein the claims do not define any structural and functional interrelationships between the software elements and a computer. Claim 1 has been amended here to further clarify the structural and functional interrelationships between the claimed compiler and a computer. In particular, claim 1 has been amended to recite a memory and that each of the operating system, the COBOL routine, and the COBOL program are stored on a computer-readable medium. Therefore claim 1 defines structural and functional interrelationships between the software elements and a computer which permit the described functionality to be realized, and is thus statutory. See *In re Lowry*, 32 F.3d 1579, 1583-84, 32 USPQ2d 1031, 1035. Dependent claims 2-15 are therefore also directed to statutory subject matter.

Claims 1-43 were rejected as directed to processes wherein the claimed subject matter does not produce a tangible result. Applicant respectfully submits that claims 1-15 are not directed to processes. Annex IV(a) of the Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility states, "When a computer

program is claimed in a process where the computer is executing the computer program's instructions, USPTO personnel should treat the claim as a process claim ... When a computer program is recited in conjunction with a physical structure, such as a computer memory, USPTO personnel should treat the claim as a product."

Assuming arguendo, Applicant respectfully submits that claim 1 does produce a useful, tangible, and concrete result. In particular, claim 1 produces the tangible result of the COBOL program receiving the address of a memory space. Applicant respectfully submits that in the broadest sense, the limitation of the COBOL program receiving the address is a tangible result of "storing" the output (i.e., the address) and is certainly is a tangible result of "conveying" the output. Further, as disclosed in paragraph 069 of the disclosure, "receiving" the address by the COBOL program produces a functional change such that "the COBOL program 12 considers the information stored in memory at the address 94 as local and accessible to the COBOL program 12. This technique is effective to enable memory queues for the COBOL programming language." Therefore, Applicant respectfully submits that claim 1 does produce a tangible output capable of being stored, displayed, or conveyed in a manner causing a useful functional change in a computer system so as to achieve a practical application. Dependent claims 2-15 are therefore also directed to statutory subject matter.

Claim 16:

With regard to amended claim 16, Applicant respectfully submits that the methods or processes do produce a useful, tangible, and concrete result. In particular, Claim 16 produces the result of resolving the memory space to an operable portion of

the COBOL program. As discussed above with claim 1, in the broadest sense, the limitation of the COBOL program resolving the memory space is a tangible result of "storing" the output (i.e., the address of the memory space) and is certainly is a tangible result of "conveying" the output. Further, as disclosed in paragraph 069 of the disclosure, "resolving" the memory space produces a functional change such that "the COBOL program 12 considers the information stored in memory at the address 94 as local and accessible to the COBOL program 12. This technique is effective to enable memory queues for the COBOL programming language." Therefore, Applicant respectfully submits that claim 16 does produce a tangible output capable of being stored, displayed, or conveyed in a manner causing a useful functional change in a computer system so as to achieve a practical application. Dependent claims 17-22 are therefore also directed to statutory subject matter.

Claim 23:

With regard to amended claim 23, Applicant respectfully submits that the methods or processes do produce a useful, tangible, and concrete result. In particular, claim 23 produces the result of sharing the block of memory. Applicant respectfully submits that "sharing" may encompass any number of tangible results such as reading data from the block of memory or writing data to the block of memory. Applicant respectfully submits that claim 23 does produce a tangible result and is therefore statutory. Dependent claims 24-30 are therefore also directed to statutory subject matter.

Claim 30:

With regard to claim 30, Applicant respectfully submits that the methods or

processes do produce a useful, tangible, and concrete result. In particular, claim 30 produces the result of receiving an address of the shared memory block. As discussed above with claim 1, the limitation of the COBOL program receiving the address is a tangible result of "storing" the output (i.e., the address) and is certainly is a tangible result of "conveying" the output. Further, as disclosed in paragraph 069 of the disclosure, "receiving" the address by the COBOL program produces a functional change such that "the COBOL program 12 considers the information stored in memory at the address 94 as local and accessible to the COBOL program 12." Also, as disclosed in paragraphs 077 and 080 of the disclosure, "receiving" the address by the COBOL program produces a functional change such that the shared block of memory is usable by the COBOL program. Therefore, Applicant respectfully submits that claim 30 does produce a tangible output capable of being stored, displayed, or conveyed in a manner causing a useful functional change in a computer system so as to achieve a practical application. Dependent claims 31-43 are therefore also directed to statutory subject matter.

Response to Rejections under Section 102

In the Office Action dated March 28, 2007 claims 1-43 were rejected under 35 USC § 102(a) as being anticipated by "BEA Tuxedo®: Using the ATMI /Q Component," January 2003, BEA Systems, Inc. (hereinafter "BEA2003").

Claim 1:

I. BEA2003 does not disclose an operating system maintaining a key and an address of the memory space related to the key.

The Office Action relied on disclosure in BEA2003, starting on page 2-4, of naming conventions used with naming queue spaces, queues, and services. Applicant respectfully submits that this section of BEA2003 does not provide any teaching or suggestion of an operating system maintaining a key and an address of a memory space related to the key as required by claim 1. A search of BEA2003 only resulted in three occurrences of discussion of an operating system. On pages 3-12 and 4-14, BEA2003 mentions an operating system in the context of an indication of a failure when enqueuing messages. In particular, indicating that "an operating system error occurred." On page A-5, BEA2003 mentions an operating system in the context of setting appropriate variables of the environment with the BEA Tuxedo system in accordance with the operating system.

II. BEA2003 does not disclose a COBOL routine maintaining the key in an index and communicating with the operating system to receive the address of the memory space.

The Office Action relied on disclosure in BEA2003, starting on page 2-4, of naming conventions used with naming queue spaces, queues, and services. Applicant

respectfully submits that this section of BEA2003 does not provide any teaching or suggestion of a COBOL routine maintaining an index including the key. Further, Applicant respectfully submits that this section of BEA2003 does not provide any teaching or suggestion of communicating with the operating system to receive the memory address based on the key in the index. Also, none of the sections of BEA2003 concerning an operating system discussed above include any teaching or suggestion of communicating with the operating system to receive the memory address based on the key in the index as required by claim 1.

III. BEA2003 does not disclose a COBOL program that communicates with the COBOL routine to receive the memory address based on the key.

The Office Action relied on disclosure of BEA2003, starting on page 4-2, of ATMI COBOL language functions for enqueuing and dequeuing messages. While BEA2003 generally teaches enabling queues for COBOL programs through the ATMI COBOL language functions, BEA2003 does not provide any teaching or suggestion of a COBOL program receiving a memory address of a memory space, such as a queue space. A search of BEA2003 only resulted in two occurrences of discussions of memory addresses. On pages 3-15 and 3-16, BEA2003 discloses that when using the C programming function tpdequeue(), an argument of the function may indicate the address of the buffer where messages being dequeued may be placed. Applicant respectfully submits that there is no teaching or suggestion in BEA2003 of a COBOL program receiving the memory address as required in claim 1.

Dependent claims 2-15 are similarly not taught or suggested by BEA2003 for at least the reasons detailed in sections I-III above.

Claim 16:

Claim 16 includes limitations similar to those presented in claim 1. Therefore, the arguments in sections I-III are herein repeated for claim 16.

Dependent claims 17-22 are similarly not taught or suggested by BEA2003 for at least the reasons detailed in sections I-III above.

Claim 23:

IV. BEA2003 does not disclose communicating a call to an operating system for a block of memory.

As discussed above in section I, BEA2003 only has three occurrences of discussion of an operating system. On pages 3-12 and 4-14, BEA2003 mentions an operating system in the context of an indication of a failure when enqueuing messages. In particular, indicating that "an operating system error occurred." On page A-5, BEA2003 mentions an operating system in the context of setting appropriate variables of the environment with the BEA Tuxedo system in accordance with the operating system. Applicant respectfully submits that BEA2003 does not provide any teaching or suggestion of communicating a call to an operating system for a block of memory as required by claim 23.

V. BEA2003 does not disclose a COBOL program communicating a request for an address of the block of memory.

The Office Action relied on disclosure of BEA2003, starting on page 2-4, of naming conventions used with naming queue spaces, queues, and services. Applicant respectfully submits that this section of BEA2003 does not provide any teaching or suggestion of a COBOL program requesting an address of a block of memory.

The Office Action also relied on disclosure of BEA2003 on page 1-5 of peer-to-peer communication. While BEA2003 does generally discuss peer-to-peer communication, there is no teaching or suggestion of a COBOL program requesting an address of the block of memory as required by claim 23. BEA2003 discusses peer-to-peer communication in the COBOL programming on page 4-30. Again, this section of BEA2003 does not provide any teaching or suggestion of a COBOL program requesting an address of the block of memory as required by claim 23.

VI. BEA2003 does not disclose returning the address to a linkage section of the COBOL program.

The Office Action relied on disclosure of BEA2003, starting on page 2-4, of naming conventions used with naming queue spaces, queues, and services. The Office Action also relied on disclosure of BEA2003, starting on page 4-2, of ATMI COBOL language functions for enqueuing and dequeuing messages. Applicant respectfully submits that neither of these sections of BEA2003 teach or suggest returning the address of the block of memory to a linkage section of the COBOL program as required by claim 23.

Dependent claims 24-29 are similarly not taught or suggested by BEA2003 for at least the reasons detailed in sections IV-VI above.

Claim 30:

Claim 30 includes limitations similar to those presented in claims 1 and 23. Therefore the arguments of sections II, V, and VI are herein repeated for claim 30.

Dependent claims 31-43 are similarly not taught or suggested by BEA2003 for at least the reasons detailed in sections II, V, and VI above.

Conclusion

Applicant respectfully submits that the present application is in condition for allowance for the reasons stated above. If the Examiner has any questions or comments or otherwise feels it would be helpful in expediting the application, he is encouraged to telephone the undersigned at (972) 731-2288.

The Commissioner is hereby authorized to charge payment of any further fees associated with any of the foregoing papers submitted herewith, or to credit any overpayment thereof, to Deposit Account No. 21-0765, Sprint.

Respectfully submitted,

Date: 6/28/2007

Michael W. Piper
Reg. No. 39,800

ATTORNEY FOR APPLICANT

CONLEY ROSE, P.C.
5700 Granite Parkway, Suite 330
Plano, Texas 75024
(972) 731-2288
(972) 731-2289 (facsimile)